Hindawi
Mobile Information Systems
Volume 2017, Article ID 6265021, 9 pages
https://doi.org/10.1155/2017/6265021

Research Article

DYMO Self-Forwarding: A Simple Way for Reducing the Routing Overhead in MANETs

Enrica Zola, Francisco Barcelo-Arroyo, and Israel Martin-Escalona

Universitat Politècnica de Catalunya (UPC), C. Jordi Girona 1-3, 08034 Barcelona, Spain

Correspondence should be addressed to Israel Martin-Escalona; imartin@entel.upc.edu

Received 2 February 2017; Accepted 11 April 2017; Published 6 July 2017

Academic Editor: Juan C. Cano

Current routing protocols in Mobile Ad hoc Networks tend to use information on the position of the nodes in order to improve their features. In fact, without this information, protocols are hardly scalable since they tend to overflow the radio media with control packets, most of them being useless at the end. This paper presents the assessment of a modification of the DYMO protocol in order to include and use positioning information. The evaluation is carried out through simulations in realistic environments and connectivity condition. The possible error in the position is seldom considered in this kind of studies but here taken into account to catch the impact of realistic GPS devices or other sources of location techniques.

1. Introduction

Mobile Ad hoc Networks (MANET) have been a hot topic for several years [1], since they came with several issues uncovered by fixed wired networks. Most of these issues are bound to the dynamic topology of MANETs. Thus, the role played by routing protocols in MANETs, in which the nodes move in an unpredictable fashion in general, has a great impact on their performance [2]. The consequences of poor designs are present mainly in metrics as relevant as the packet delay, overall overhead, CPU load and memory consumption (hence energy consumption and lifetime), and so forth. MANETs tend to share the same issues regardless of the specificity of the network. Thus, in Vehicular Ad hoc Networks (VANETs), where vehicles act as network nodes, routing protocols have to deal with nodes that move fast and abruptly leave or join the network [3]. The same applies to Wireless Sensor Networks (WSN), where nodes can be either static or expected to move freely [4]. In any case, the introduction of new sensors in WSN must be easy, and since it involves changes in the network topology, again routing protocols are key to keep the performance metrics within the desired bounds. In disaster-recovery scenarios [5], the network infrastructure is considered unreliable, which usually involves the network working with very limited resources, often set up by the rescue team members. Under these conditions, the network topology can change abruptly anytime and anywhere. Accordingly, the performance of the routing protocol becomes essential, since it has to address the data delivery duty using weak routes that frequently break [6].

In the past, a wide variety of routing protocols have been proposed. Those protocols that ignore the position of the nodes are in general inefficient since they need some degree of flooding. Techniques based on flooding are goaled to establish neighborhood relationships in a quick way, and from these relationships, the routing strategies are derived. If the network topology does not change or if the changes are slow, the overhead caused by flooding control packets can be kept at acceptable levels. However, fast changes in the network topology, caused by the movement or by the introduction of new nodes, bring the overhead to unacceptable figures; after the overhead, other metrics also worsen.

This paper presents a modification of the DYnamic MANET On-demand (DYMO) routing protocol [7] that includes the position of the node in the routing strategy. This modification was previously introduced in [8] where simulations were also included showing improvements for some simple cases. Here, further details are shown about the implementation and results of the proposed modification.

The main goal of this work is to assess the feasibility of the proposed modification in scenarios with different node densities, where the impact of the routing overhead due to the flooding in DYMO may be critical. The main original contributions that differ and improve our previous study in [8] are as follows: (1) a discussion on the Dynamic Forward Delay (DFD) function is presented along with the formulation developed for our routing proposal; (2) the scenarios proposed here include scenarios with better connectivity (higher density of nodes), which are aimed at better modeling of the conditions in which VANET and WSN often perform; (3) there are further details on the development of the simulation tool; (4) results presented here include second-order statistics to illustrate stability; and (5) this paper also presents cases in which the position of the node is known with a certain degree of error showing how robust the algorithm is in front of the typical lack of position accuracy in devices that must not be too complex. As typically done by other authors [9, 10], this work assumes that the position of the node is known through a system external to the routing procedure (typically GPS). Although the positioning technique could be embedded within the routing protocol and make use of the already necessary control packets [11], this is left for further research.

The remainder of this paper is organized as follows. The routing strategies that are related to the proposed protocol are described in Section 2. Section 3 provides the details of the proposed modification on DYMO. The simulation tool and the functions developed for the new algorithm are detailed in Section 4, together with the simulation settings considered in our evaluation. The results of the performance evaluation of the proposed protocols are presented in Section 5. Finally, Section 6 concludes the paper.

2. Related Routing Strategies

2.1. AODV. The Ad hoc On-Demand Distance Vector (AODV) routing protocol exchanges control routing packets among the nodes of the network whenever a node is willing to send data to another node for which the route is not known. The route is then temporarily stored in routing tables for further use in the near future and deleted after a given amount of time if unused. A Route Request (RREQ) is broadcasted through the network during route discovery; any node that receives this RREQ and does not have a known route to the destination has to rebroadcast the RREQ. If an intermediate node has a route to the destination or if the destination node is reached, a Route Response (RREP) is sent to the source node and the routing path is created and stored (i.e., any intermediate node receiving the RREP creates forward route entries for the destination in the routing table).

Hello packets and link layer feedback is used in order to maintain routes in the table: a route error (RERR) packet is sent to the source node whenever an intermediate node detects a link break on that route, so that the source node initiates the route discovery process again in order to update the route.

Many authors have already discussed the heavy routing overhead and complexity problems with regard to implementation of the AODV protocol [12–14].

2.2. DYMO. The DYMO (recently renamed as AODV-v2 [15]) is a reactive routing protocol that does not send control packets unless it is performing routing or transmitting tasks. It routes hop by hop. If a node needs to transmit and there is some known route to the destination, it just uses it before anything else. If a route is not kept or the stored route fails, the node starts the procedure of route discovery by broadcasting messages of RREQ in which the node's address is included. Once the destination node receives the RREQ it replies with a RREP message addressed to the source node; once the RREP is received at the origin the route is established and available in both ways [16].

The main difference between DYMO and AODV is that the former allows each intermediate node in the route between a source and a destination to store the route to all the predecessor nodes, thus lessening the routing overhead.

Basically DYMO does not maintain routing tables after topology changes, thus avoiding sending a number of control packets otherwise useless if specific transmissions are not needed for a given topology change. The monitoring of routes only takes place when traffic is available for transmission. However, the nodes check the status of the links with their neighbors through a set of timers. Details on these timers and how they work can be found in [15].

2.3. Beaconless Routing (BLR) Protocol. The Beaconless Routing Protocol (BLR) is a position-based routing protocol that does not need to periodically broadcast Hello messages or create and maintain routing tables [17]. When a source node S needs to send data to a destination node D, it will broadcast the data directly. Among all its neighboring nodes, only one node will be selected in a distributed manner by means of restricting the forwarding to a subset of nodes (i.e., those in the forwarding area) and by applying a forwarding delay (i.e., a function of the node's position with respect to D and S); the position information of D and S is carried in the data packet. The neighbor node inside the forwarding area (i.e., a circle with diameter r relative to the previous forwarding node in the direction of the destination node D) and with the smallest forwarding delay will be the one that will rebroadcast the data first. The rest of the candidate forwarding nodes will then avoid retransmission, thus lessening the number of packets in the network.

According to [17], several delay functions, called Dynamic Forwarding Delay (DFD), can be used in BLR. For instance, the DFD can then be computed as in MFR [18]:

$$\mathrm{DFD} = D_{\max} \cdot \frac{r - p}{r}, \tag{1}$$

where r is the maximum transmission range, p is the progress of the node towards the destination (i.e., the projection of the distance traveled over the last hop onto the line from previous node to destination), and $D_{\max}$ is the maximum delay a packet can experience per hop. In this way, the number of

hops is minimized [19]. A different strategy can be used when nodes are able to adjust their transmitting power, as proposed in [20]. In that case, making the node with the least progress to relay the packet can minimize the energy consumption; that is,

$$\mathrm{DFD} = D_{\max} \cdot \frac{p}{r}. \quad (2)$$

More recently, authors in [21] applied similar delay logic to the message relaying in Vehicular Ad hoc Networks (VANETs). Authors in [22] demonstrated that exponentially distributed timers can further decrease the number of responses compared to uniformly distributed timers. Moreover, more advanced DFD functions can be drawn that take into account the distance to the previous node [17].

3. Including Position Information to DYMO

In this section, the details of the proposed modification on the DYMO routing protocol are provided. The main idea behind our proposal is to merge the benefits of the selective forwarding strategy promoted by the BLR approach and the benefits of the route discovery and routing table maintenance of the DYMO protocol. To this end, it is assumed that the position information of the sending node and of the destination node is known.

3.1. General Description of DYMOselfwd. The knowledge of the coordinates where each node is actually located can drastically improve the performance of routing protocols in ad hoc networks. In fact a reasonable degree of scalability is very hard or even impossible to achieve without this knowledge since the need for control packets increases more than linearly with the increase of the number of nodes.

The full description of the proposed protocol, called DYMOselfwd (DYMO with selective forwarding) was presented in [8]. The proposed modification applies to the retransmission of the RREQ in DYMO and is aimed at reducing the routing overhead in the network. First of all, the RREQ message is slightly modified in order to include two fields: (1) the position of the sending node and (2) the position of the destination node. The sending node at each hop changes the former, while the latter is kept unchanged during the route discovery. These two fields are necessary for the node at each hop to calculate distances, as described in the following section. According to our proposal, the source node needs to know the destination node's position, while all the nodes in the network are required to know their own position.

When a node receives a RREQ for which the route is unknown, it first checks whether its distance to the destination node (owndist) is smaller than the distance of the node that sent the RREQ to the destination node ($\mathrm{owndist}_{\mathrm{prev\_node}}$). If so, it is a candidate forwarding node for the RREQ and it further processes the RREQ; otherwise it drops the message. This first step already decreases the number of relaying nodes, thus reducing the routing overhead. Although the Euclidean distance is not always the best choice, as it does not guarantee converging to the destination, we recall that this work is focused on scenarios with high density of the nodes. The performance of DYMOselfwd in low-density scenarios has been already covered in [8], where it has been shown that flooding techniques (e.g., DYMO) perform better. As future work, it is intended to address again scenarios with poor connectivity where, in the case of route failures, we may switch from one routing protocol (e.g., DYMOselfwd) to the other (e.g., DYMO).

Borrowing the idea from the BLR protocol, each candidate forwarding node has to wait a given time (i.e., DFD) before it can resend the RREQ. This delay is a function of its distance from the destination node: the closer it is, the smaller the delay is. In this way, the candidate node closer to the destination will be the first one in resending the RREQ, while the other candidate nodes, after overhearing this new RREQ with a higher sequence number, discard the pending retransmission. Again, limiting the number of relaying nodes helps reducing the routing overhead in the network; moreover, it aids preserving battery power at the nodes and avoids interference with regular data transmissions.

3.2. Dynamic Forward Delay. The waiting time that each candidate forwarding node has to wait before transmission of the RREQ is computed as a function of its distance from the destination node (owndist). An exponentially distributed timer is used to further decrease the number of RREQ [22]. In this work, the DFD has been tuned in order to cope with internal timers and with the event execution order in OMNeT++. To this end, a minimum processing time ($D_{\min}$) at each node needs to be considered, which leads to the following DFD function:

$$\mathrm{DFD} = D_{\min} + (\alpha - 1) \cdot 10 \cdot D_{\min}, \quad (3)$$

where $D_{\min}$ is set to 126 μs and α can be computed as

$$\alpha = \frac{e^{\mathrm{owndist}/1000}}{e^{d_{\max}/1000}}. \quad (4)$$

owndist is the distance between the destination node and the node that has received the RREQ; $d_{\max}$ is given by

$$d_{\max} = \mathrm{owndist}_{\mathrm{prev\_node}} - r, \quad (5)$$

where $\mathrm{owndist}_{\mathrm{prev\_node}}$ is the distance from the destination node of the node that previously sent the RREQ and r is the maximum distance allowed in order to correctly receive data packets (i.e., 71 m in this work). The denominator in (4) aims at proportionally reducing the introduced delay at each step, as an absolute delay (i.e., the numerator in (4)) may lead to very high delays in wide scenarios.

4. Simulation Tool

The OMNeT++ was selected as an adequate tool to carry out simulations in order to show a first insight into the performance of the DYMO protocols without and with the proposed modification. OMNeT++ [23] is a discrete event simulator that provides a modular structure where the

models are built with existing modules that can be combined in different ways. Several developments with different versions of DYMO were tested and discarded since they were not useful for implementing the desired modifications. The development reported in [24] was selected since after the tests it proved to be adequate for implementing DYMOselfwd. This selected module corresponds to version 10 of the draft [16]. Several upgrades have been implemented in order to properly work as described in the draft and to allow the achievements of statistics:

(i) Messages sent by DYMO have been labeled sequentially in order to distinguish the first try from resendings.

(ii) Time stamps have been added to some events that did not have them in the original implementation. Also counters for DYMO messages have been added.

(iii) Broken routes due to moving nodes are detected by means of a trigger at the MAC layer. After 7 consecutive ACK are lost for the same packet, the route is labeled as broken.

(iv) A route error message (RERR) is sent to the source node when a route is labeled as broken so it can try to establish a new route.

4.1. Modified Functions for DYMOselfwd. Besides the previous modifications that were necessary in order to guarantee a behavior in line with the draft, other functions have been added or modified for the implementation of DYMOselfwd protocol:

(i) *receiveChangeNotification()*: this function enables the DYMO module to take actions when given notifications are received, thus allowing the implementation of the modifications proposed by DYMOselfwd.

(ii) *calculateDistance()*: this function calculates owndist for the given node. The output is a float that is used by the node for deciding whether it is a candidate node and for calculating the DFD.

(iii) *fwdNode()*: this function compares owndist with the distance between the previous sending node and the destination (owndistprev_node) (we recall that owndistprev_node is included in the received RREQ packet) and determines whether the node is a candidate for retransmission of the RREQ.

(iv) *waitingTime()*: this function calculates the DFD, that is, the delay that the candidate node has to wait before eventually retransmitting the RREQ.

(v) *checkSeqNum()*: this function checks whether a new RREQ for the same destination and with a higher sequence number has been received by the node during the waiting time.

(vi) *addOwndist()*: this function modifies the sending node field in the RREQ packet by substituting owndistprev_node with owndist.

Figures 1 and 2 depict the flow chart for the DYMO and DYMOselfwd, respectively. In both protocols, when a node receives a message, it first determines whether it is a control message through the *handleLowerRM()* function: if not, the packet is discarded (i.e., data packets are not processed by the routing protocol, which is only responsible for determining the route to follow); otherwise it is passed to *myAddr()* function which checks whether the node is the destination of the message. If true, a RREP is sent to the source host through the functions *handleLowerRMForMe()* and *sendReply()*; if not, the two protocols have different behavior. DYMO resends the received message (i.e., a RREQ or a RREP).

On the other hand, DYMOselfwd first checks whether the received message is a RREP, in which case it is resent through the functions *handleLowerRMForRelay()* and *sendReply()*. Otherwise, it has to calculate owndist and determine whether it is a candidate forwarding node through the new function *calculateDistance()* (the new functions implemented in OMNeT++ are in italics in Figure 2); if so, a delay is calculated based on owndist through the *waitingTime()* function and a timer is started; otherwise the message is discarded. After the timer has expired, the node checks whether a new RREQ has been received with a higher sequence number through the function *checkSeqNum()*, in which case the node discards the message. Otherwise, it first modifies the RREQ packet by changing $\text{owndist}_{\text{prev\_node}}$ with owndist (implemented by the new function *addOwnDist()*) and then resends the RREQ through the functions *handleLowerRMForRelay()* and *sendRequest()*.

4.2. Simulation Settings. As pointed out in [25], evaluating routing protocols with different topologies, densities, and mobility characteristics is important in order to outline possible weaknesses in the ad hoc routing process. To this end, two basic scenarios have been used for testing purposes with 13 and 25 nodes, respectively. In [8] similar results are presented for cases with poor connectivity considering 9 nodes within the same simulation area, where the strategy is expected to perform poorly as also stated in [17]. The geographic area simulated here corresponds to a square shape of 150 × 150 m meters. Each node has a coverage range of 71 m. Although simple, the selected simulation area is intended for assessing the expected reduction in the routing overhead in DYMOselfwd, which is the main purpose of this work. As a bigger area would incur in longer paths where the nodes mobility may hazard the stability of the routes, this is left for a future study once the feasibility of the proposed protocol has been consolidated. The mobility model selected is the Random WayPoint (RWP) without pause times. Each result is obtained after averaging the results of 5 independent simulations of the same case, each simulation running along 1,250 s of simulation time. With this simulation setting, the confidence interval is better than 95% in all cases. Other parameters necessary for the simulation have been configured as usual in other similar works [25, 26] and are summarized in Table 1.

The two layouts investigated are represented in Figure 3. In fact, this layout is used for the static cases and as the

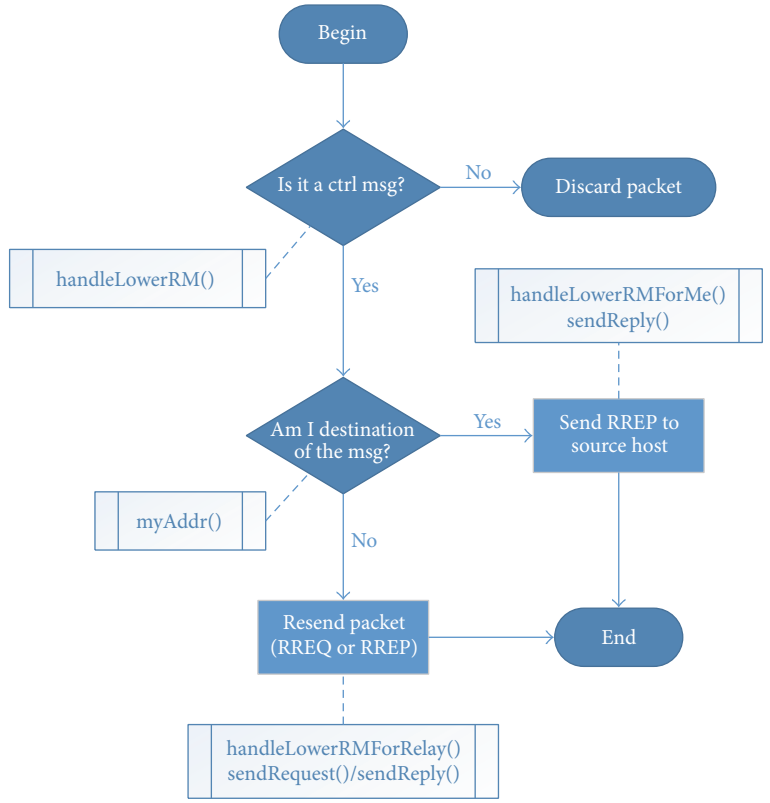

Figure 1: DYMO protocol and its corresponding OMNeT++ functions.

Table 1: Configuration parameters used in simulation.

Simulation time	1,250 s (×5)
Area size	Squared 150 m × 150 m
Number of nodes	13/25
Mobility model	Random Way Point
Average speed	1 and 2 m/s
Routing protocol	DYMO/DYMOselfwd
Propagation model	Two Ray Ground Model
Coverage radius	71 m
Shadowing	0 dB
Transport protocol	UDP
Frame size	512 Bytes
Sending rate	1 frame every 250 ms
Protocol MAC	802.11 g
Bit rate	54 Mbps
Frequency band	2.4 GHz

starting layout in case of moving nodes that are afterwards left to move according to the random model selected. The source node (*TX*) and the destination node (*sink*) are represented in black, while the candidate forwarding node (*best node*) is colored in grey. In the high density scenario, due to the symmetry, two *best nodes* are present. The coverage area is represented in red. The owndist of the *best node* in each scenario is also depicted.

The metrics selected to evaluate the performance of both algorithms are the following:

(i) Troute: time to establish the route (when the source receives the first RREP from the sink)

(ii) ColMac: number of collisions at MAC level

(iii) MACPdu: total number of MAC frames sent by all nodes

(iv) REQtot: total number of RREQ sent by all nodes

Other metrics have been also collected with the goal of tracking the simulation. The number of packets sent by the source has been compared to the amount received by the sink and in all cases they agree within a negligible margin due to the possibility of some packets still traveling when the simulation stops or to a negligible probability of loss.

5. Simulation Cases and Results

5.1. Static Cases. In these cases, the nodes are assumed to remain static along the whole observation period according to the layout displayed in Figure 3. The results displayed in Table 2 include the average and the standard deviation. All the metrics displayed are better for DYMOselfwd except the *Troute* in normal density, which is slightly higher (1.5% higher). The standard deviations are all low as it could be expected from a static topology and contrary to what will be presented below for moving nodes.

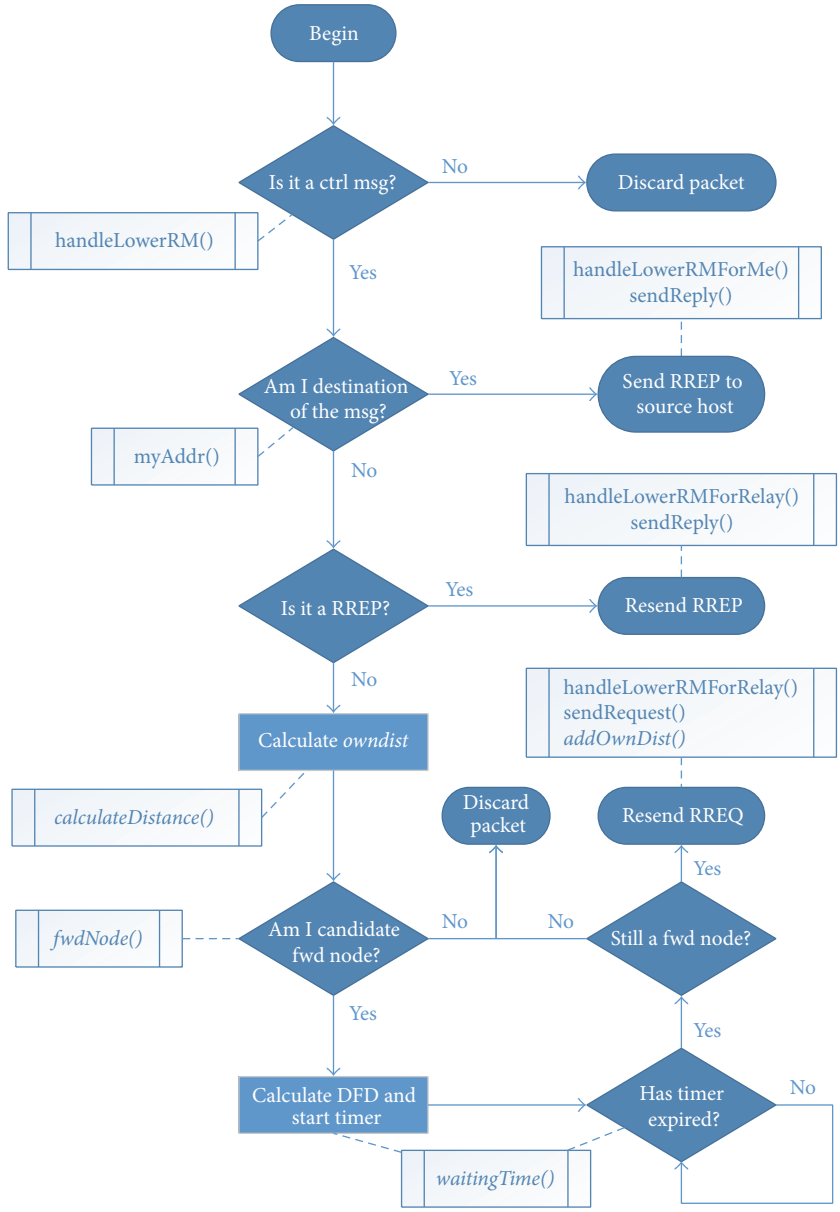

Figure 2: DYMOselfwd protocol and its corresponding OMNeT++ functions. New functions are in italics.

Table 2: Results for static nodes (average and standard deviation).

	Normal density		High density	
	DYMO	*DYMOselfwd*	*DYMO*	*DYMOselfwd*
Troute [ms]	5.71/0.9	5.79/0.9	4.77/0.8	4.43/0.7
ColMac	30.2/3.7	0.0/0.0	392.4/36.0	5.8/11.0
MACPdu	20,351/5	20,176/13	15,688/111	15,191/2
RREQtot	265.6/4	84.0/0	555.6/6	128.0/2

Table 3: Other results on the RREQ in the static scenario.

		Normal density	High density
RREQtot/*N*	*DYMO*	20.43	22.22
RREQtot/num_hop	*DYMOselfwd*	21.00	2×21.33
RREQtot/REQ_TX	*DYMO*	12.65	26.46
	DYMOselfwd	4.00	2×3.05

Table 3 shows that, in the static scenario, the number of RREQ in DYMO is proportional to the number of nodes *N*, while in DYMOselfwd it is proportional to the number of *num_hops* (recall that in the high density scenario there are two *best nodes* retransmitting the RREQ). Notice that twice the amount of RREQ is sent due to the perfect symmetry of the layout and the fact that nodes keep static at the same place along all the simulation time. In addition, the number of *REQtot* versus the number of the RREQ transmitted by *TX* (*REQ_TX*) is always lower in DYMOselfwd; moreover, in DYMO it increases when the nodes' density increases,

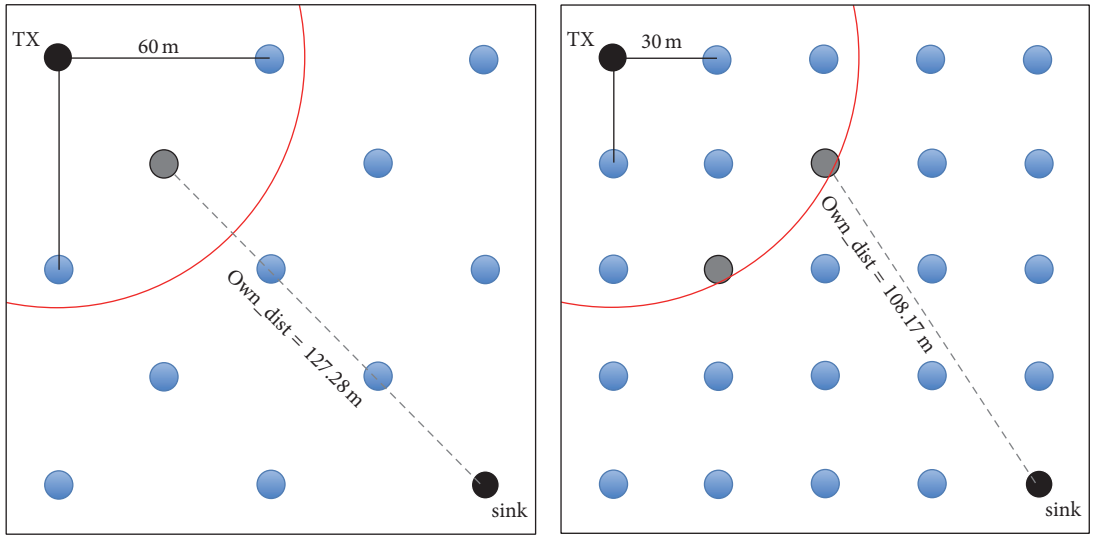

Figure 3: Nodes' locations in the simulation area for the 13 (normal density) and 25 (high density) nodes scenarios. *TX* is the node sending UDP segments to the *sink*. Its coverage area is represented in red.

Table 4: Results for normal speed (average and standard deviation).

	Normal density		High density	
	DYMO	*DYMOselfwd*	*DYMO*	*DYMOselfwd*
Troute [ms]	4.23/0.8	5.90/0.8	4.25/0.9	4.56/0.6
ColMac	1,186/896	213/33	9,215/2,618	725/184
MACPdu	20,684/2,588	16,060/499	24,262/1,468	16,732/177
RREQtot	2,247/864	527/63	4,056/573	570/59

while it decreases in DYMOselfwd unless two *best nodes* are retransmitting.

5.2. Moving Nodes. If we assume that nodes move, we can expect a decrease in quality caused by the well-known mobility cost phenomenon. Movement involves continuous changes and hence the need to discard and rebuild routes and the extra need for overhead introduced by these tasks.

First, we present the case with a normal pedestrian speed of 1 m/s on average, for which results are displayed in Table 4. First the severe impact of mobility on ColMac must be noticed, which experiences a drastic increase from the static case. The *Troute* is worse for the DYMOselfwd while all other metrics remain much better with the new strategy. This is because, in the static case, the node at the upper corner was always at sight of two others, while now because of the random movement this amount of two can change: while an increase does not improve a lot the time to establish the route, reducing the nodes at sight to none has a noticeable impact. A diffusion-like strategy as the one implemented by DYMO is of course faster. However, results prove that it is not much faster while the impact on the number of collisions and other metrics is noticeable. Now the standard deviations (and also their quotient to the average, i.e., coefficient of variation) are much higher than for the static case, and this is another consequence of the mobility. With a changing topology, the metrics also change more between different simulations. However, the better stability of DYMOselfwd that reduces not only the metrics but also their coefficient of variation must be stressed: for instance, the *ColMac* for normal density reduces by a factor of 5 and the standard deviation by 27.

The results for fast speed of 2 m/s are presented in Table 5. Results are in general worse than for normal speed. Some results that seem to be better now show very high standard deviation involving an important degree of possible error (lack of stability in the simulations). The trends are the same when increasing the speed: DYMOselfwd decreases the overhead and is more stable; a higher density of nodes makes these improvements more obvious.

5.3. Position Error. In the results presented we assumed a perfect knowledge of the nodes' positions (i.e., with no error). In general, this is not the case and nonnegligible errors will be present in the nodes' positions. To assess the performance of DYMOselfwd under nonideal positioning we simulated a scenario in which the actual node's positions are impacted by an unbiased Gaussian error, with a root mean squared error of about 5 meters. This error is typical in assisted GPS receivers [27, 28].

The scenario with 25 nodes and medium speed was simulated obtaining the results displayed in Table 6. Notice that result for DYMO with position error makes no sense since DYMO does not use positions. The results presented

Table 5: Results for fast speed (average and standard deviation).

	Normal density		High density	
	DYMO	*DYMOselfwd*	*DYMO*	*DYMOselfwd*
Troute [ms]	4.81/0.5	5.90/0.8	4.26/0.9	4.56/0.6
ColMac	982/771	413/64	14,581/4,073	1,671/136
MACPdu	19,540/2,822	16,893/722	29,039/3,383	18,346/180
RREQtot	2,409/1,001	907/90	6,741/848	1,105/69

Table 6: Results with position errors.

	DYMO	DYMOselfwd no errors	DYMOselfwd error
Troute [ms]	4.25	4.55	4.41
ColMac	9,215	725	842
MACPdu	14,262	16,732	17,089
REQtot	4,056	570	582

here are only slightly worse than those presented when the location error was absent and better than in DYMO. The main conclusion of this experiment is that the proposed protocol behaves well under small errors in the nodes' positions.

6. Conclusions

While routing protocols that do not take into account the location of the nodes tend to send a big and unnecessary amount of control packets, the inclusion of location allows this amount to be drastically reduced. The modification of DYMO presented here keeps the main features of DYMO while obtaining a noticeable reduction of control packets. The consequence is a reduction of collisions and hence an improvement of the available throughput.

The results presented show how DYMOselfwd performs well in the tested scenarios and is quite robust to changes in speed and connectivity, notwithstanding its simplicity. The consequence of errors on performance of DYMOselfwd is also small proving that the protocol is adequate to realistic scenarios where the location is known with a degree of error.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The authors would like to express their gratitude to Jordi Ruiz Paños and Raúl Ruiz Díaz who were in charge of implementing the simulations. This research was supported by the Spanish Government and ERDF through CICYT Project TEC2013-48099-C2-1-P.

References

[1] M. G. Rubinstein et al., "A survey on wireless Ad Hoc networks," in *Mobile and Wireless Communication Networks. IFIP The International Federation for Information Processing*, G. Pujolle, Ed., vol. 211, Springer, Boston, Mass, USA, 2006.

[2] S. Taneja and A. Kush, "A survey of routing protocols in mobile Adhoc networks," *International Journal of Innovation, Management and Technology*, vol. 1, no. 3, August 2010.

[3] F. Li and Y. Wang, "Routing in vehicular ad hoc networks: a survey," *IEEE Vehicular Technology Magazine*, vol. 2, no. 2, pp. 12–22, 2007.

[4] J. N. Al-Karaki and A. E. Kamal, "Routing techniques in wireless sensor networks: a survey," *IEEE Wireless Communications*, vol. 11, no. 6, pp. 6–28, 2004.

[5] V. G. Menon, J. P. Pathrose, and J. Priya, "Ensuring reliable communication in disaster recovery operations with reliable routing technique," *Mobile Information Systems*, vol. 2016, Article ID 9141329, 2016.

[6] V. M. Patel and N. N. Gondaliya, "Energy efficient ad-hoc routing protocol for disaster scenario," *International Journal of Computer Applications*, vol. 98, no. 18, pp. 43–49, July 2014.

[7] C. Perkins, Dynamic MANET On-demand (AODVv2) Routing, February 2013 https://tools.ietf.org/html/draft-ietf-manet-dymo-26.

[8] E. Zola, F. Barcelo-Arroyo, and I. Martin-Escalona, "A modification of DYMO routing protocol with knowledge of nodes' position: proposal and evaluation," in *Proceeding of the 13th International Conference on Wired/Wireless Internet Communications*, vol. 9071, pp. 289–298.

[9] E. G. D. P. Raj, S. Selvakumar, and J. R. Lekha, "LBRP: geographic routing protocols for MANETs," in *International Conference on Recent Trends in Information Technology, (ICRTIT '11)*, pp. 318–323, June 2011.

[10] M. Ayash, M. Mikki, and K. Yim, "Improved AODV routing protocol to cope with high overhead in high mobility MANETs," in *Proceedings of the 6th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, (IMIS '12)*, pp. 244–251, July 2012.

[11] I. Martin-Escalona, M. Malpartida, and F. Barcelo-Arroyo, "Performance evaluation of the passive TDOA algorithm in dark areas," in *Proceedings of the 2012 Ubiquitous Positioning, Indoor Navigation, and Location Based Service, (UPINLBS '12)*, pp. 1–8, Helsinki, Finland, October 2012.

[12] D.-W. Kum, J.-S. Park, Y.-Z. Cho, and B.-Y. Cheon, "Performance evaluation of AODV and DYMO routing protocols in MANET," in *Proceedings of the 2010 7th IEEE Consumer Communications and Networking Conference, (CCNC '10)*, pp. 1–2, Las Vegas, Nev, USA, January 2010.

[13] A. Goel and A. Sharma, "Performance analysis of mobile ad-hoc network using AODV protocol," *IETECH Journal of Communication Techniques*, vol. 4, no. 1, pp. 6–11, 2010.

[14] R. Kochher and R. Mehta, "Performance analysis of reactive AODV and DSR with Hybrid GRP Routing Protocols under IEEE 802.11g MANET," in *Proceedings of the International*

Conference on Wireless Communications, Signal Processing and Networking WiSPNET '16, pp. 1912–1916, Chennai, India, March 2016.

[15] C. Perkins, "Ad Hoc On-demand Distance Vector Version 2 (AODVv2) Routing," January 2016, https://tools.ietf.org/html/draft-ietf-manet-aodvv2-13.

[16] I. Chakeres and C. Perkins, "Dynamic MANET On-demand (DYMO) Routing draft-ietf-manet-dymo-10," IETF Internet-Draft, 2008, IETF Internet-Draft, 2008, 2008, http://tools.ietf.org/search/draft-ietf-manet-dymo-10.

[17] M. Heissenbüttel, T. Braun, T. Bernoulli, and M. Wälchli, "BLR: beacon-less routing algorithm for mobile ad hoc networks," *Computer Communications*, vol. 27, no. 11, pp. 1076–1086, 2004.

[18] H. Takagi and L. Kleinrock, "Optimal transmission ranges for randomly distributed packet radio terminals," *IEEE Transactions on Communications*, vol. 32, no. 3, pp. 246–257, 1984.

[19] T. Braun, M. Heissenbüttel, and T. Roth, "Performance of the beacon-less routing protocol in realistic scenarios," *Ad Hoc Networks*, vol. 8, no. 1, pp. 96–107, 2010.

[20] T.-C. Hou and V. O. K. Li, "Transmission range control in multihop packet radio networks," *IEEE Transactions on Communications*, vol. 34, no. 1, pp. 38–44, 1986.

[21] M. De Felice, E. Cerqueira, A. Melo, M. Gerla, F. Cuomo, and A. Baiocchi, "A distributed beaconless routing protocol for real-time video dissemination in multimedia VANETs," *Computer Communications*, vol. 58, pp. 40–52, 2015.

[22] J. Nonnenmacher and E. W. Biersack, "Scalable feedback for large groups," *IEEE/ACM Transactions on Networking*, vol. 7, no. 3, pp. 375–386, 1999.

[23] OMNeT++, Discrete Event Simulator. https://omnetpp.org.

[24] C. Sommer, I. Dietrich, and F. Dressler, "A simulation model of DYMO for ad hoc routing in OMNeT++," in *In Proc. of the 1st Int. Conference on Simulation tools and techniques for communications, networks and systems workshops*, Article 76, 8 pages, Simutools '08. ICST, Brussels, Belgium, 2008.

[25] C. Sommer, I. Dietrich, and F. Dressler, "Simulation of Ad hoc routing protocols using OMNeT++ : a case study for the DYMO protocol," *Mobile Networks and Applications*, vol. 15, no. 6, pp. 786–801, 2010.

[26] E. Zola and F. Barcelo-Arroyo, "Impact of mobility models on the cell residence time in WLAN networks," in *Proceedings of IEEE Sarnoff Symposium (SARNOFF '09)*, Princeton, NJ, USA, March 2009.

[27] S. Savasta, M. Pini, and G. Marfia, "Performance assessment of a commercial gps receiver for networking applications," in *Proceedings of the 2008 5th IEEE Consumer Communications and Networking Conference, (CCNC '08)*, pp. 613–617, Las Vegas, Nev, USA, January 2008.

[28] M. Modsching, R. Kramer, and K. ten Hagen, "Field trial on GPS Accuracy in a medium size city: The influence of built-up," in *Proceedings of the 3rd Workshop on Positioning, Navigation and Communication (WPNC '06)*, pp. 209–218, WPNC'06, Bremen, Germany, October 2016.